Boston University
Electrical & Computer Engineering
EC464 Senior Design Project

Final Report
(Version 2)

IoT Kitchen

Submitted to:
Annie Lane
262-444-3207
aelane@bu.edu

by

Team 21
IoT Kitchen

Team Members
Addison Dolido addison@bu.edu
Erin Dorsey edorsey1@bu.edu
Saransh Kothari saranshk@bu.edu
Yuran Shi yuran@bu.edu
Kenny Zheng ykzheng2@bu.edu

Submitted: April 10 2020

Table of Contents

Executive Summary

Team 21 - IoT Kitchen

Many amateur chefs often face difficulty when trying to follow along with recipes and updating recipes to their preference in real time. In addition, there are many existing technologies to aid in cooking but none that concentrate them all into an easy to use hub. IoT Kitchen seeks to solve these issues as the ultimate smart cooking assistant for amateur home chefs. IoT Kitchen provides hands free assistance so that chefs can follow recipes as well as update recipes to their own preferences in real time. This is achieved with the help of a user friendly app interface, Natural Language Understanding (NLU) assistant, as well as several IoT sensors including a scale, thermometer, and measuring cups. The Android app interface will allow users to keep a record of their recipes made on their profile page, and find new recipes by viewing other users' profiles. If a user fails to find inspiration from their friends, they can search our curated recipe database with strategic recipe keywords such as ingredient or dietary restriction. When a user finds a recipe, they can initialize auditory recipe instructions through the NLU assistant. This assistant will help a user step through a recipe or jump to a specific ingredient or instruction. If the user comes across an element of the recipe that they'd like to change, they can vocalize this change to the NLU assistant, which will send that information back to the recipe database and app. Throughout cooking, the IoT sensors will track cooking time, temperature, and ingredients added, providing feedback to the app to update the recipe and to correct a user if they are following the recipe wrong. Combined, IoT Kitchen will make following along to recipe and cultivating new ones easier than ever before.

1.0 Introduction

If you are like most people, you do not look forward to the cooking process. Many amateur chefs often face difficulty when trying to follow along with recipes and updating recipes to their preference in real time. Although there are many existing technologies to aid in cooking, none concentrate specifically all into an easy to use hub. This is why IoT Kitchen was created. With smart kitchen appliances and an android mobile application, you will save time in the kitchen. These futuristic appliances explain the steps of the cooking process, analyze cooking preferences, and make food preparation much more efficient.

IoT Kitchen is a smart kitchen assistant that utilizes an IoT scale, temperature sensor, RFID measuring cups as well as natural language processing to break down the barriers for amateur cooks. IoT Kitchen makes it easier than ever for amateur homecooks to make recipes by using our appliances in conjunction with the mobile application to sort out what ingredients are being put into a dish in addition to having the natural language processing assistant give the user instructions as they are following a recipe in order to simplify the cooking process.

Our approach to create IoT Kitchen consists of the following components: Hardware, Natural Language Understanding (NLU), and Mobile Application. In our hardware portion of this project, we plan on creating a scale, thermometer, and RFID measuring cups which will all relay information to the mobile application. Our mobile application recipe categories will be catered to chefs based upon their dietary preferences. For instance, if a user wishes to find dishes with chicken, there will be a category specifically for chicken related recipes. This nifty tool displays visual recipes in a step-by-step manner and helps you measure the perfect amounts for your catered recipes. If you decide to alter the recipe or if you do not have an ingredient, this gadget will adjust the other ingredients as appropriate or provide alternative choices for suitable substitutes. Lastly, if a user wishes to change a specific ingredient, they are able to do so with the NLU assistant which will send that information back to the recipe database and mobile application which then documents the changes. The IoT sensors will track cooking time, temperature, and ingredients added and will relay this information to the mobile application to update the recipe and provide guidance to the user as to whether or not they are following the steps of the recipe correctly.

Each of the smart kitchen appliances above facilitates cooking making it that much easier for the average person to create tasty dishes in surprisingly little time. More importantly, the ease of preparing nutritious meals in a smart kitchen will encourage people to eat healthy homemade dishes as opposed to resorting to unhealthy fast foods. Ultimately, IoT Kitchen will not only shorten the cooking process but will promote and support people to consume nutrient dense dishes.

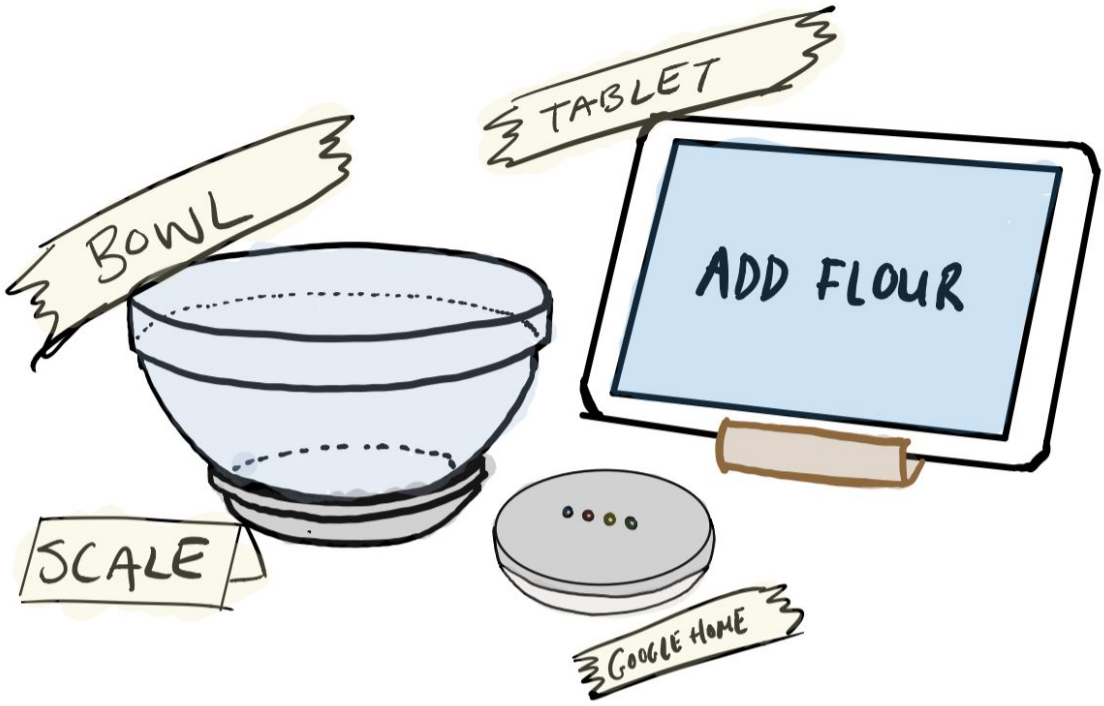


Figure 1. Concept Drawing of IoT Kitchen Implemented in a Kitchen

2.0 System Technical Description

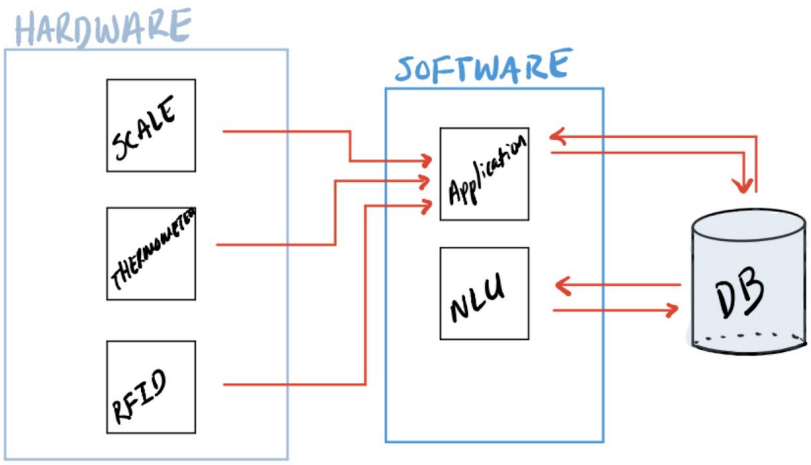


Figure 2. Whole System Block Diagram
The three hardware sensors send data to the software application that, alongside the Natural Language Understanding software module, reads and writes from the database

Hardware

The hardware elements for this project are various sensors that collect data about a user's cooking session. Initially there will be three unique sensors, two of which will transmit measured data to the android app via bluetooth. The data these sensors collect will be viewable by the cook to see more detailed information about the recipe they created, such as the weight of flour used or the temperature of the chicken versus time. These three sensors are described individually below.

The first sensor is a smart scale that will measure the weight of ingredients used. This scale is made up of an arduino uno, a 5kg strain gauge load cell, an HX711 load cell amplifier, an HC-05 bluetooth module, an LCD screen, and two buttons. All of this will be contained in a water resistant 3D printed housing to allow for splashing and cleaning. The cook will be able to place a bowl onto the scale and press a button to tare the scale. Then when the instructions say to add 50 grams of flour, the scale can measure that the cook actually used 53 grams of flour. When the cook is done adding that ingredient they can push another button to send that data to the Android app and continue onto the next recipe instructions. The scale will be calibrated to measure to the nearest gram and will be sensitive enough to measure 1 gram. The wiring of this sensor is shown in Figure 3 below:

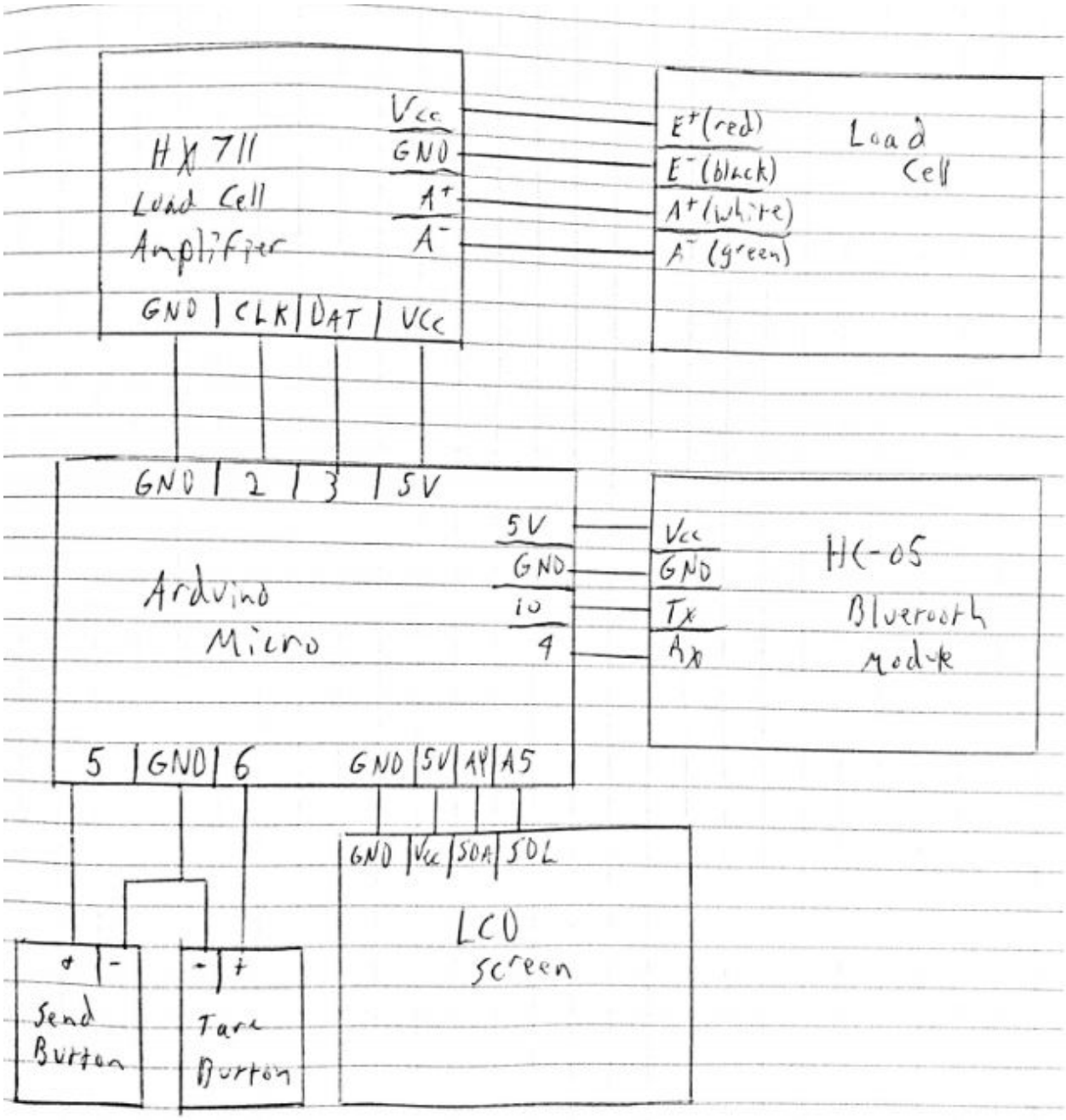


Figure 3. Circuit schematic for the scale.

The next sensor is an oven temperature sensor that will measure the internal temperature of food inside the oven, such as a chicken. This is made up of a type-k glass braid insulated stainless steel tip that can measure up to 700°C and is food and oven safe which is then connected to a MAX31856 thermocouple amplifier that works with temperatures well above 700°C. All of this is also attached to an arduino uno and an HC-05 bluetooth module. The cook will be able to stick the thermocouple into the food item going into the oven and then attach the

rest of the components onto the outside of the oven, so that the bluetooth signal receives no interference and the hardware does not overheat. This will allow the cook to tell when their food has reached the proper internal temperature and is ready to be removed from the oven. The wiring of this sensor is shown in Figure 4 below.

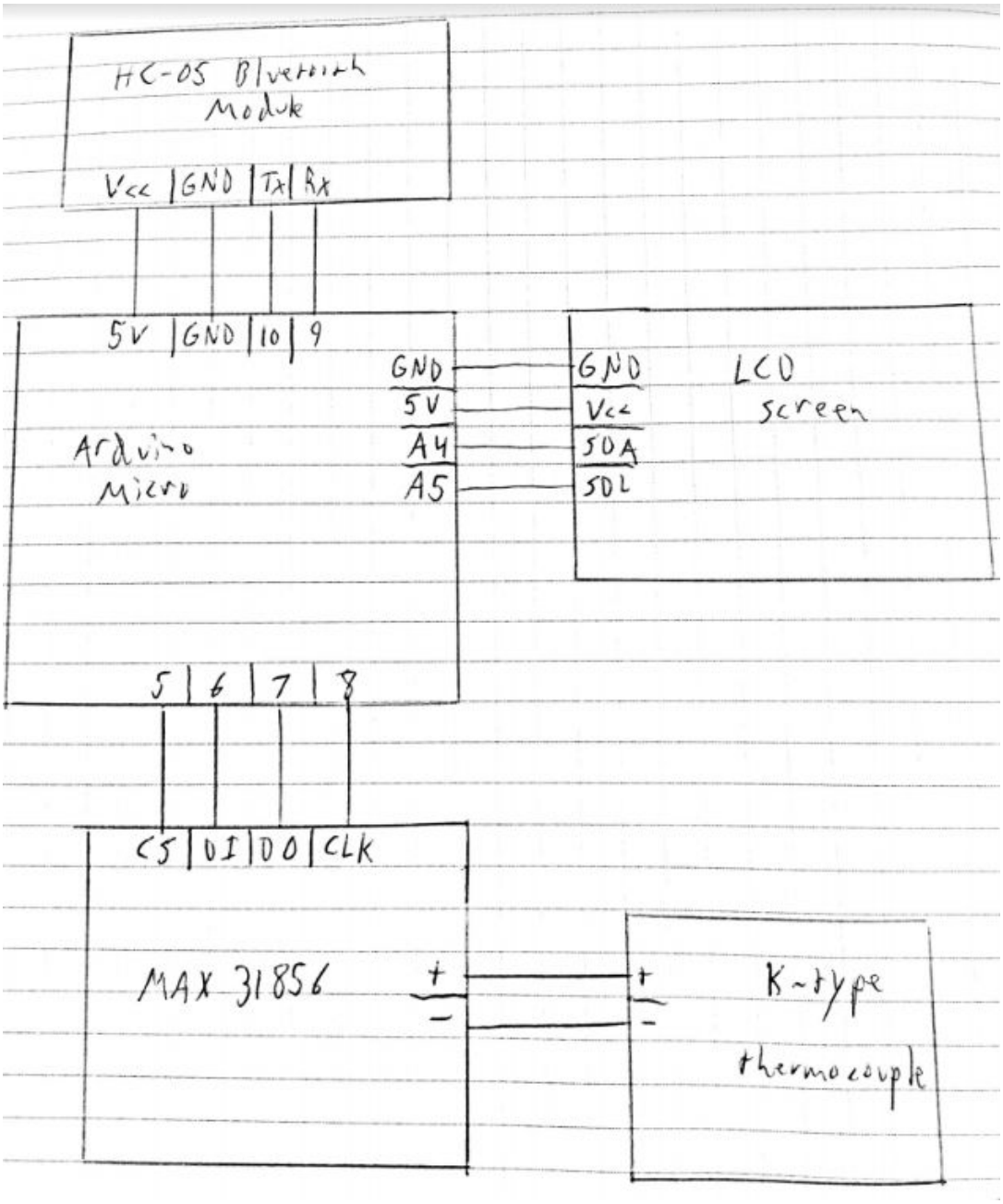


Figure 4. Circuit schematic for the temperature sensor.

Finally, we will utilize RFID tags to aid the user in downloading and getting started with the mobile application as well record the volume of ingredients used through RFID tags that will

be placed on measuring cups. The RFID tags will be created by combining an RFID chip, antenna and encapsulating it within an appropriate substrate. These tags will be passive and will function in the high frequency range (13.56 MHz) so that an android smartphone may be used as an RFID reader by utilizing its NFC protocol. The antennas used will consist of either a copper or aluminium coil with 3-7 turns. By utilizing NFC tags, a subset of High Frequency RFID tags, we will ensure that the tags have a read range of 3 cm or less. This will aid the user in ensuring that an RFID tag is scanned only when intended.

Software Mobile Application

The software component of this project acts as a hub that connects to the hardware components HC-05 module via bluetooth and Wifi, displaying data on the tip of your hands. This portion of the project is what allows users to store and update recipes and search for recipes based upon different preferences. Our main sections of the mobile application includes the following: Home, Profile, Recipes, and Settings all using Android Studio, Java and Google Firebase.

In the Home section, users are presented with the newsfeed which will include suggestions based upon the time and/or day. For instance, if it is the morning, there will be recipe suggestions such as omelets with bacon. Following the Home page will be the users very own profile page which will include favorite recipes, things they have cooked, calendar, saved recipes and total time it took them to cook for that specific day.

The Profile page consists of user authentication which is for users to login to their specific data by using Google Firebase. Users are able to display their favorite dishes as well as dishes they have previously cooked.

The Recipes page is catered based upon the users diet and will include the following: chicken, beef, low-carb, high-protein, vegan, etc which is called from the MealDB API. By doing so, users are able to search and filter out what recipes work for them. This will then navigate them to the preview recipes page before they start cooking which includes ingredients, recipe instructions, weight and amount. Once the user wishes to start cooking, they will be shown a page of instructions which will help guide them through each instruction through the utilization of our NLU. Lastly, we plan on including a settings tab that can sync and configure devices such as the Google Home. This is also where the bluetooth connectivity with the Arduino UNO is made possible via Bluetooth.

Figure 5. Preliminary GUI of Mobile Application

Natural Language Understanding

The Natural Language Understanding (NLU) elements of the project allow a user to follow and interact with a recipe through voice commands and auditory responses. The user can interact with the NLU agent through a Google Smart device such as an Android phone enabled with Google Assistant or a Google Home. The agent works by receiving voice commands from a user and using Google's Dialogflow NLU API it can match specific words and commands to an intent. Intents correlate to specific functions and processes that the agents will execute. Additional specificity comes in the form of entities, which are variables for intent functions that specific keywords are matched to. For example, the word "first" is matched to the stepNum entity for the intent to find a specific instruction or ingredient. Then, fulfillment code written in javascript is used to connect the Dialogflow agent to a Google Firebase database containing the user's recipe information. This code connects the agent to the database via a webhook request and either writes or reads to the database in a different function for each intent. This fulfillment returns parts of the recipe to Dialogflow for the agent to speak aloud depending on the input of the user.

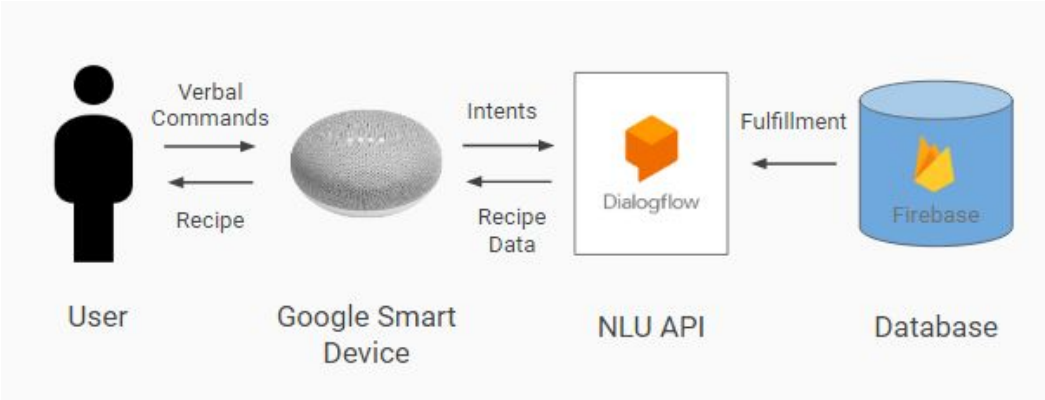


Figure 6. Block Diagram of NLU Agent
A user speaks a command to the Google Smart Device which is interpreted by the Dialoglow NLU API. Data is retrieved from Firebase via fulfillment and recipe data is conveyed to user.

The functionality of this NLU agent is twofold. A user can utilize this agent to follow along with a recipe by asking for ingredients or instructions, or a user can instruct the agent to change elements of a recipe. First, a user initializes the agent by saying “Talk to IoT Kitchen” to whatever Google smart device they prefer to use. Then they specify their unique user code that is provided to them by the IoTKitchen app. Once the agent knows what user’s recipes to access, the user can vocalize the code of whatever recipe they would like to make. This initializes the agent to pull from the database page of that recipe. To follow a recipe, a user can ask for the title, details (servings and time information), ingredients, or instructions. They can specify which ingredient or instruction they would like, such as “first”, “second”, “last”, etc.. Alternatively, the user can step through the recipe with commands like “next”, “back”, or “repeat”. These queries are matched to fields in the recipe database page and returned to the user. If the user would like to change an element of the recipe at any point they must specify whether they would like to change an ingredient, instruction, or detail, and which number they would like to change. These specifications will match the user command to a key in the recipe database page. Then, the user can supply the alteration to the recipe text and the agent will send a script to automatically update that value in the recipe database with the new text. When updating an instruction, a user can specify that they would like to update the corresponding ingredient along with their changes to the instruction. This makes it easy to save all changes that a chef makes in real time.

3.0 Second Semester Progress

Hardware

The work done on hardware this semester can be broken down into three areas, one for each sensor being built, the scale, temperature sensor, and RFID tags.

The scale was nearly complete from last semester, meeting our goals for both sensitivity and functionality. In regards to actual hardware, two physical buttons were added to the scale for the purpose of tarring and sending the data to the app. The main work that was done this semester was designing a waterproof housing for the components. This included researching the best materials for this purpose and then designing the housing to be made. We concluded that the fastest, and cheapest method would be the most optimal, which was 3D printing the housing and then filling all openings with waterproof glue in some areas and gaskets for areas that were used in the actual weighing of the ingredients. This was fully designed in Solidworks, but was unable to be manufactured due to the semester being cut short.

The second sensor is the temperature sensor, which began this semester as mainly an idea with some components bought. Throughout the semester we assembled all of the sensor components, consisting of an Arduino micro, HC-05 bluetooth module, LCD screen for testing, MAX31856 Universal Thermocouple Amplifier, and a K-type thermocouple with glass braid insulation and a stainless steel tip. We had picked these components with the idea that everything would hang outside of an oven door while the thermocouple snaked through the door and into the food item to measure its temperature. With this put together, we tested and tweaked to ensure that the thermocouple could withstand and read the highest temperatures most ovens can produce ~600 F and would not block the door. All of our tests were successful with no noticeable heat loss from any ovens tested when the sensor was snaked through the door and the temperature was safely read all the way up to around 550 F. The final work for the temperature sensor was creating a housing for the thermocouple so that it could be stuck into food items safely. Through extensive research we had decided to use stainless steel, the same material as the tip of the thermocouple, which is food safe. We designed and began to manufacture this housing, but was interrupted once again due to the Coronavirus.

The final sensor is the RFID tag system. NFC tags have been written to correspond to the following sizes of measuring cups: 1 cup (250 ml), ½ cup (125 ml), ⅓ cup (80 ml), ¼ cup (60 ml), and ⅛ cup (30 ml). These NFC tags have been attached to the bottom of each corresponding measuring cup. There are two primary reasons for this occurrence. Firstly, it is imperative that there is enough area for the NFC tags to be placed on the cups without bending the antenna within the tags. Secondly, the placement of the NFC tags will enable the user to scan the tags using an android device with greater ease. The measuring cups used are made from stainless steel for practical purposes. By using stainless steel we can be certain that the measurements on the cups will not wear off during the washing process. Furthermore, the cups will be stamped with both imperial and metric units allowing for easier conversion. The final component of the RFID

system is an NFC tag that has been written with the aim of assisting the user in downloading the mobile application and to provide a brief intro to the application and its utility. This is so that the user will have a smoother initial experience with the mobile application and may start reaping the benefits of what it can provide more swiftly. Extensive research was conducted on how to integrate the written NFC tags with the mobile application based on the existing android RFID documentation. However, due to a shortened semester we were unable to assimilate the NFC tags with the mobile application as envisioned.

Although bluetooth connectivity is an important part of both the scale and the temperature sensor, that progress is more software related and is discussed in the section below titled “Software Mobile Application”.

Software Mobile Application

Extensive work was done in regards to the mobile application this semester. The team was able to strike a balance between aesthetic and functionality. In terms of front end development, we implemented visual elements that users interact with and see. This includes the User Log-In Page, along with User Profile, Recipes, Instructions and Scale. Images were used for the background. We used Sketch to create the tasteful effects. In particular, we uptaked material templates and elements consisting of fonts, texts, backgrounds, and color, with intentions to modify and fix bland looks. This allowed us to create a mobile friendly, and modern mobile application.

Now for the backend portion of the project, we decided to rework our current code in order to make it more organized, readable and cleaner while limiting bugs and crashes. This entailed creating new classes for specific functions, such as one for the Database, Bluetooth, and Login. All of the appropriate code was moved to more reasonable areas whilst making it more efficient. During this phase, the software team configured the bluetooth functionality to be a service rather than part of an activity, meaning that it would be able to run in the background on any page the user may navigate to. While several crashes also occured due to invalid page navigation and logging out; all of these crashes were fixed with the navigation bar not causing any crashes.

After improving and condensing our code, the team began to enhance some of the pages, increasing their functionality and retrieving data from the database. On top of that was connecting the database and welcoming the user by their name. However, the main parts of this phase was the creation of model classes used to hold all relevant recipe information. This allowed a single database call to retrieve and populate a variable with all of the current recipes and relevant information, such as the ingredients, steps, time to cook, etc... With all of this information gathered and easily accessible we revamped several of the screens to provide more information on every recipe. This mainly affected the recipe select screen which now displays the time it takes to cook the recipe and how many servings it will make.

Afterwards came retrieving the bluetooth data from the sensors and then storing all relevant data to be viewed later by the user. This entailed making the bluetooth service work properly, connecting and receiving the relevant information from the sensors. With bluetooth working we needed to parse the information and then store it locally until the recipe was finished at which point all relevant data is stored in the database (the weights and the corresponding ingredient, the temperature vs. time, and finally a reference to the recipe itself).

Lastly, we coded a navigation bar for the application in order to navigate through the application. This allowed the user to login with options of going to the following menus: scale, user profile, signout, or recipe list. The front-end of the navigation bar was simply editing the background, font, color etc. We also added final touches to pages. In terms of the recipe page, besides making it pleasing to look at, once the user clicks what recipe they wish to cook, they will be prompted to the instruction/ingredients page. We also added the option to go back and next on the specific recipe. At this stage the semester was cut short and so although data is stored in the database successfully, there is currently no way to view the data in the app.

Natural Language Understanding

The majority of the semester has been spent developing and testing the functionality for a user to change their recipe through the Natural Language Understanding agent. This consisted of training the agent and writing fulfillment code for the changeServe, changeTime, changeIngredient, and changeInstruction intents. For the changeInstruction intent, 5 follow-up intents were also developed, including Ingredient, Number, Procedure, Weight, and Update. These intents were built using the agent training functions on the Dialogflow console and writing fulfillment code in javascript to connect to the Google Firestore Cloud Database. The fulfillment for these intents differed from the intents developed during the first semester because they required functions that would write to the database rather than just reading from it. This caused challenges with indexing into the database and influenced the decision to restructure the recipe database.

The database restructuring was needed in order to standardize how the NLU and Software teams utilized the database for their respective functions. The format of the ingredient and instruction fields in the database were changed from a nested array format to a map of strings to allow for easier indexing. In addition, the instruction field was changed to a single string to several string fields that separate ingredient, number, procedure, unit, and weight. This allowed for data to be pulled from the database to connect to the smart scale while simplifying the changeInstruction procedure. To finish up the database restructuring, fulfillment code for each intent was updated for the new indexing and tested for bugs and errors.

The next element of progress this semester was to perform comprehensive debugging and error checking. User error is somewhat unpredictable, but the NLU team attempted to estimate all the ways a user could break either the agent or the fulfillment code and write error checking code to make sure any incorrect input would not break the NLU module. Using Dialogflow,

checks and fallback responses were built into each intent to reprompt the user if any input was given that could not be mapped to an intent or did not provide the correct parameters for that given intent. In the fulfillment code, checks were written to avoid over-indexing the database or using any variables that were unassigned due to missing parameters in the intent. To aid with this error checking, more comprehensive instructions were written into the agent's reponses to help guide the user to the correct input. During prototype testing the agent was able to match all user input without any errors, and during informal testing incorrect user input was not able to crash the module.

The final progress made this semester was taking steps towards deployment. A test deployment was executed early in the semester, but the in progress Action was rejected for release by the Google Team. The reasons for rejection included unclear user instructions and a missing privacy policy. In response, the NLU team updated and elaborated the user instructions both in the Action description and within the intents themselves. A simple privacy policy was also written.

4.0 Technical Plan for Project Completion

Hardware - Scale

Task 1. Buy all of the required components and assemble prototype circuit
All of the parts making up the scale are bought and assembled using temporary wires and breadboards for testing purposes. The correct hookup of all of the components can be seen in figure 3.
Lead: Addison
Duration: 2 Weeks

Task 2. Write the Arduino code
With the circuitry made, write the arduino code for the scale with the ability to tare, send data via bluetooth, and view the current weight on the LCD screen. The code can be found in our github.
Lead: Addison
Duration: 1 Week

Task 3. Create a temporary load cell housing
Using wood or some other material, attach the load cell using screws between two slabs. One slab is then against the table and the other is where the weights are placed. Make sure to use spacers between the wood and the load cell so that the load cell is not directly against either surface.
Lead: Addison
Duration: 1 Week

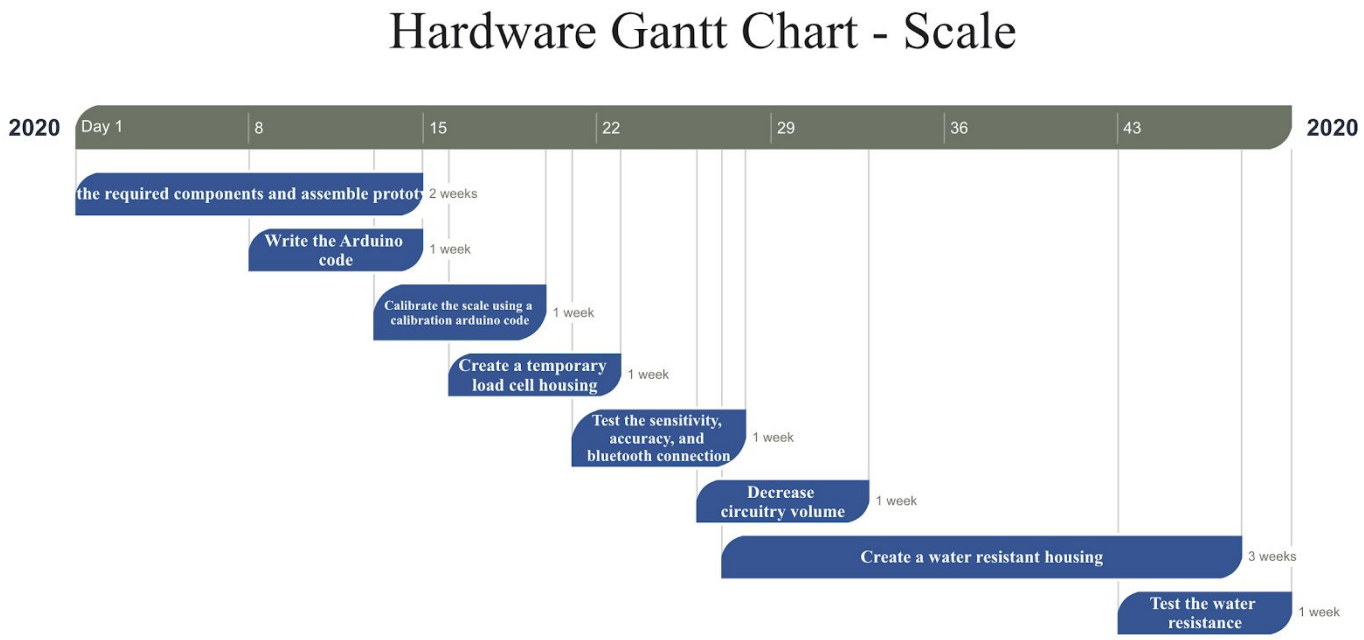
Task 4. Calibrate the scale using a calibration arduino code
The load cells used measure weight by a linear equation y=mx+b. The library used automatically calculates b, but the slope y must be found. This can be done by placing a known weight on the scale and recording what the output y is. Then one can plug in to y=mx+b (b is made to be 0) and solve for m.
Lead: Addison
Duration: 1 Week

Task 5. Test the sensitivity, accuracy, and bluetooth connection
At this point the scale prototype is in working order. Test the scale to ensure that it is sensitive to within one gram, can measure a single gram, and can connect to a mobile device via bluetooth.
Lead: Addison
Duration: 1 Week

*Task 6. Decrease circuitry volume
All of the parts making up the scale should be soldered together to ensure stability and decrease the volume that they take up. If necessary a PCB can be built to replace some of the parts to further decrease volume and decrease power consumption.
Lead: Addison
Duration: 1 Week

*Task 7. Create a water resistant housing
A 3D printed casing for the internal circuitry of the scale should be designed and built to ensure the scale is water resistant. The secondary goal of the casing is to increase the overall aesthetics and useability of the scale. One can 3D print the housing, using gaskets and waterproof glue to seal up any holes to make it liquid resistant.
Lead: Addison
Duration: 3 Weeks

*Task 8. Test the water resistance
Lightly splash the scale and confirm that the scale still functions properly.
Lead: Addison
Duration: 1 Week



Hardware - Temperature Sensor

Task 1. Buy all of the required components and assemble prototype circuit
All of the parts making up the temperature sensor are bought and assembled using temporary wires and breadboards for testing purposes. The correct hookup of all of the components can be seen in figure 4.
Lead: Addison
Duration: 2 Weeks

Task 2. Write the Arduino code
With the circuitry made, write the arduino code for the temperature sensor to display the current temperature on the LCD screen and to send data over bluetooth at a rate of 1 Hz once the temperature is above 30 C. The code can be found in our github.

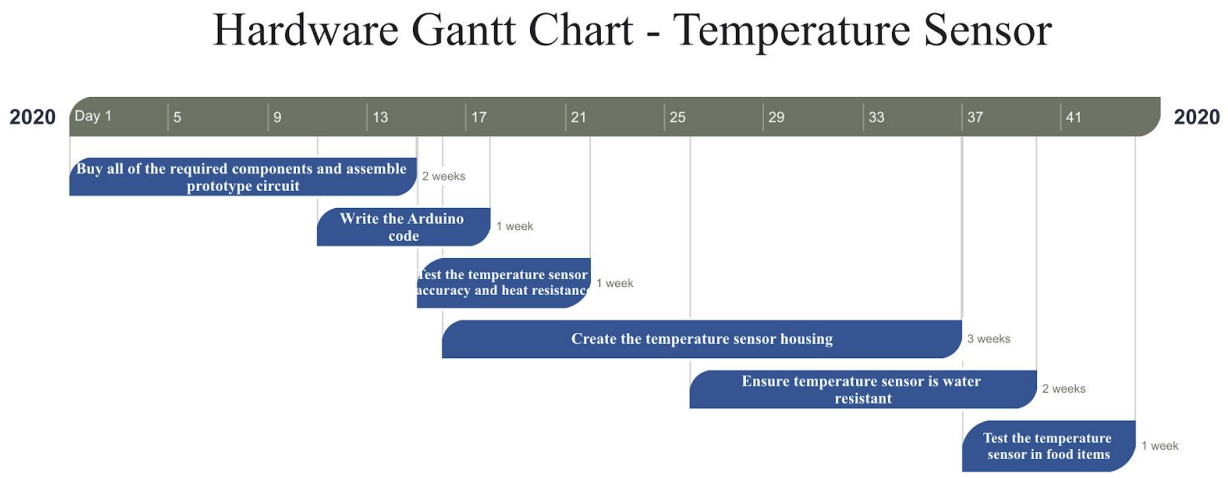
Lead: Addison
Task Duration: 1 Week

Task 3. Test the temperature sensor accuracy and heat resistance
Place the temperature sensor in an oven that displays the temperature and turn the oven on to a high heat. Compare the oven reading and the sensors reading. Confirm the correct temperature reading and check for any noticeable heat loss from the oven.
Lead: Addison
Task Duration: 1 Week

* Task 4. Create the temperature sensor housing
A housing for the temperature sensor should be designed and built to withstand oven temperatures and be food safe. The material of choice is stainless steel to match the thermocouple tip and fulfills all requirements.
Lead: Addison
Task Duration: 3 Weeks

*Task 5. Test the temperature sensor in food items
The temperature sensor with it's housing will be tested by measuring the internal temperatures of different food items throughout baking. The reported temperature will then be compared to that of a commercial food thermometer. The ease of use in getting the sensor in and out of the food will also be noted.
Lead: Addison
Task Duration: 1 Week

*Task 6. Ensure temperature sensor is water resistant
The circuitry of the temperature sensor should be soldered to decrease volume and a PCB can be built if necessary to further decrease volume and power consumption. A housing will then be created to ensure that the hardware is water resistant and to increase both the ease of use and the aesthetic.
Lead: Addison
Task Duration: 2 Weeks



Hardware - RFID

*Task 1. Add clear film on NFC tags that are attached to measuring cups
The NFC tags are attached to the bottom of each measuring cup. A clear and thin plastic film will be overlaid on each NFC tag to prevent them from coming into contact with water or any other liquid. Given that the measuring cups are made from stainless steel, protecting the NFC tags will ensure that the whole system is water resistant.
Lead: Saransh
Task Duration: 1 week

Task 2. Test water resistance of RFID measuring cups
It will be imperative to ensure that the NFC tags are in fact protected by the clear plastic film. The measuring cups will be placed in a dishwasher for one cycle with 'normal' settings applied. The condition of the plastic film and NFC tag will be inspected to ensure that no damages have occurred. Additionally, the readability of the NFC tags will be tested again to ensure they work after the cups have been washed.
Lead: Saransh
Task Duration: 1 week

*Task 3. Map mobile application download link to NFC tag that contains brief introduction to the app
The mobile application download will be mapped to a URL. The URL will then be written onto the NFC tag that already includes a text-to-speech file introducing the user to IoT Kitchen. This will result in the NFC tag containing multiple actions. When scanned, the NFC tag will need to automatically begin downloading the mobile application and during the process will relay to the user the brief introduction that is already written in the tag.
Lead: Saransh
Task Duration: 1.5 weeks

Task 4. Test NFC tag that begins mobile application download and provides brief introduction
The NFC tag will be scanned using an android device. Once scanned, within one minute the mobile application should begin downloading for the user. Whilst the mobile application is downloading, it will be important to check that the brief introduction is being played to the user as intended through the android device.
Lead: Saransh
Task Duration: 1 week

*Task 5. Integrate RFID measuring cups with the mobile application
The NFC tags on the measuring cups will need to indicate to the mobile application which specific measurement amount was scanned and this will be done by ensuring that the app is able to read the NDEF data that is stored in the tag. An intent filter will be coded into the mobile application making certain that when the NFC tags are scanned, the app will automatically capture the NDEF data without the user directing the android device to do so. The mobile application will be coded to filter for the intent ACTION_NDEF_DISCOVERED as the specific measurements are written on the NFC tags as text files with MIME type of text/plain. After the mobile application is able to process the NDEF data on the NFC tags, the app will need to be further coded to act on the data and register the specific measurement of the cup that has been scanned in order to make changes to the recipe.
Lead: Saransh
Task Duration: 3 weeks

Task 6. Test the RFID measuring cups
All the NFC tags will be individually scanned by an android device. It will be noted whether or not the mobile application is able to interpret the NDEF data and register the specific measurement amount of the cup that is scanned. It will also be tested whether or not the mobile application is able to interpret the NDEF data from the NFC tag and make changes to a recipe according to the specific cup that is scanned.
Lead: Saransh
Task Duration: 1.5 weeks

Software

Task 1. Retrieve recipe data from the database and display such data
Write code to retrieve recipe information from the database. This includes the recipe names, instructions, ingredients, etc. The recipes must then be able to be shown to the user in two separate situations:

1. When the user is choosing a recipe to cook, display all possible recipes.
2. When the user is cooking a recipe, display the current step and all pertinent information.

Lead: Addison
Task Duration: 1 Week

Task 2. Set up the bluetooth service
Create a service that runs in the background of the app that connects to both the temperature sensor and scale to receive data. The bluetooth service should be able to parse the incoming information and decide

whether it is temperature data or weight data. The service should also store all data while the recipe is in progress.
Lead: Addison
Assisting: Yuran
Task Duration: 1 Week

Task 3. Save cooking session data
Create code that saves all cooking session data in the database, such as ingredient weight and temperature, once the user has finished the recipe.
Lead: Addison
Task Duration: 1 Week

*Task 4. Test the retrieval and storage of the sensor data
Start the test up, connect one or both of the sensors and begin a recipe. Measure some ingredients at requested times and finish the recipe. Confirm that the correct weights and temperatures were stored in the database.
Lead: Addison
Task Duration: 1 Week

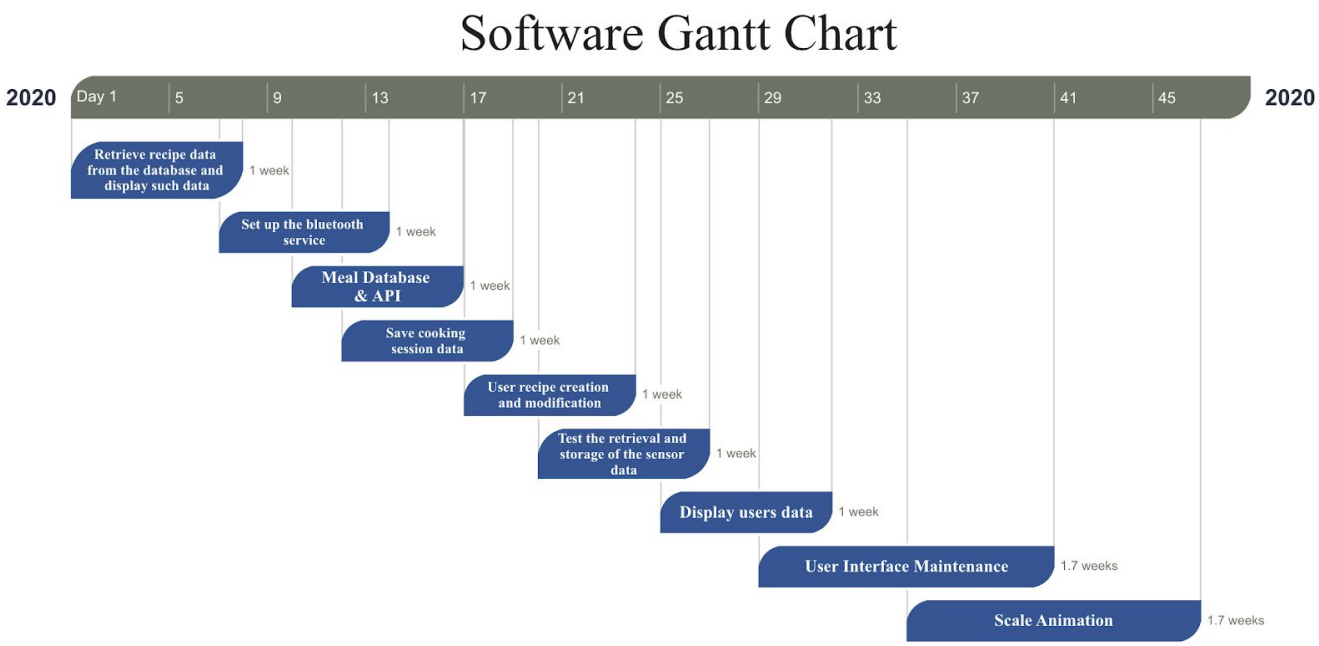
*Task 5. Display users data
Create a page to display a user's cooking session data, such as the actual weight of ingredients used, the date, a graph of the temperature, etc.
Lead: Kenny
Task Duration: 1 Week

*Task 6. Allow user recipe creation and modification
Create a page to allow the user to create a new recipe and modify an existing recipe. The new recipe is then saved in their personal user collection in the firebase database.
Lead: Yuran
Task Duration: 1 week

*Task 7. Meal Database & API
Integrate more meal recipes into Google Firebase from meal database APIs. Users should be able to edit their recipes which will be stored on their user profile.
Lead: Kenny
Assisting: Yuran
Task Duration: 1 week

*Task 8. Front End Development for Scale Animation
Finalize and integrate animations for scale and recipe instructions. Users will be able to see their progress on the scale with animations.
Lead: Kenny
Task Duration: 1.7 week

*Task 9. User Interface Maintenance
Putting together the segments of the application starting from Home all the way to User Profile and Signout. We want to fix any bugs when navigating through the app. Lastly, we hope to have a flow and user friendly interface.
Lead: Kenny
Task Duration: 1.7 week

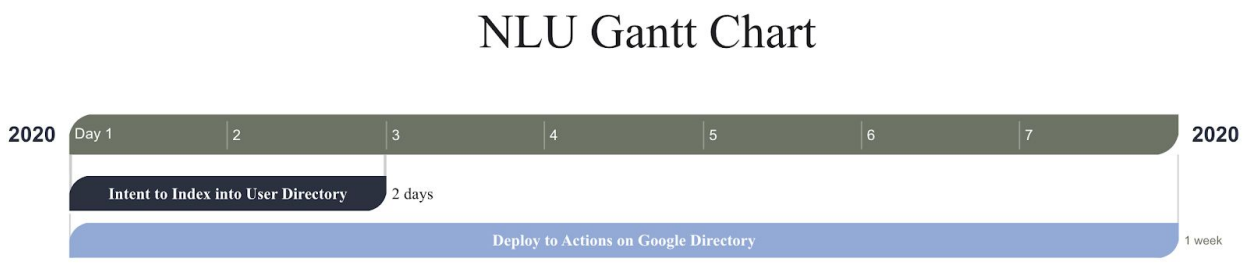


Natural Language Understanding

*Task 1. Intent to Index into User Directory
The final change to the functionality of the Natural Language Understanding module required for the completion of this project is to update the fulfillment and intents to index into each user's personal recipe database. This would require an additional intent to be formulated that asks the user for a unique user ID generated by the IoT Kitchen Android App. Then, a fulfillment script should be written that matches this user ID to the user's personal recipe database, from which all data for subsequent intents should be pulled from. For each intent's fulfillment functions, the assignment for the variables dialogflowAgentRef and dialogfrlowAgentDoc will need to be changed to reflect this new way to index into a user's database. Once this code is written it should be tested extensively by speaking all the training phrases to the agent along with a few incorrect phrases to test the accuracy of the intent.
Task Duration: 2 days (1 for writing code, 1 for testing)

Task 2. Deploy to Actions on Google Directory
In order to make the IoT Kitchen Google Action accessible to users, it needs to be deployed to the Google Actions on Google Directory so that it can be loaded onto user's smart devices. This requires going to the Actions on Google development console (https://console.actions.google.com/) and updating the Project

Description and Privacy Policy to reflect the new changes of how a user accesses their personal database. Then, the project must be released and submitted to Google for review. The review process will take at least a few days, depending on how many review requests are submitted to Google at that time.
Task Duration: 1 week (1 day for deploying, and 1 week for release to be approved)



* Tasks being worked on or planned at the time we were instructed to stop the project due to the coronavirus.

5.0 Cost Breakdown

Item	Description	Cost
1	Google - Home Mini (1st Generation) - Smart Speaker with Google Assistant ***Donated**	$20.00
2	GL.iNET GL-MT300N-V2 Mini Travel Router ***Donated**	$20.50
3	Sketch 1 Year License - Student Discount	$52.84
4	NFC ANTENNA FOR IOT - Abracon LLC	$3.20
5	MIFARE Classic 1K Dia 25mm adhesive RFID Sticker NFC tag	$14.98
6	Adafruit Micro NFC/RFID Transponder - NTAG203 13.56MHz [ADA2800]	$6.21
7	Arduino Uno	$18.50
8	2 x HC-05 Bluetooth Module	$9.98
9	Arduino Micro	$20.70
10	5kg Strain Gauge Load Cell & HX711 Load Cell Amplifier	$14.95
11	2 x 16x2 LCD Display	$12.44
12	Thermocouple Type-K Glass Braid Insulated Stainless Steel Tip	$9.95
13	MAX31856 Universal Thermocouple Amplifier	$17.50
14	10mm stainless steel rod	$22.10
15	Bellemain Stainless Steel Measuring Cup Set, 6 Piece	$18.95
	Total	$262.80

6.0 Requirements

Scale Sensitivity

Value, Range, Tolerance, Units: Can measure a change of 1 gram

Test Results:

Known weight (g)	Measured weight (g)	Percent difference (%)
500	500	0
520	520	0
521	521	0
571	571	0
71	71	0
21	21	0
1	1	0
0	0	0

Table 1: Scale weight test. The first row is the first measurement, and then weight is either added or removed for the subsequent measurement (the next row).

As shown in table 1 above, the scale accurately read a change of 1 gram, both going from 0 to 1 and 520 to 521 g. In addition, the scale returned to a reading of 0 once the last gram was removed, indicating that our scale can accurately measure a change of 1 gram.

Scale Precision

Value, Range, Tolerance, Units: Measures accurately to the nearest gram

Test Results: As shown in Table 1 above, the scale continuously measured the correct weight to the nearest gram.

Changes: Changed from 0.01g to 1g based off of research into the weights of cooking ingredients. An example is 1 cup of lettuce, which weighs over 30 grams.

Scale Taring

Value, Range, Tolerance, Units: Scale returns to 0 after taring and accurately measures weights to the nearest gram.

Test Results:

Known weight (g)	Measured weight (g)	Percent difference (%)

50	50	0
20	20	0
500	500	0
1	1	0
2	2	0
500	500	0
520	520	0
20	20	0

Table 2: Scale Taring test. First row is the first test and every subsequent row is after. Each test began with a taring of the current weight and then addition of a new weight to test the accuracy.

The taring functionality of the scale works, whenever tared, the scale returns to reporting a weight of 0. Also as shown in Table 2 above, the scale continues to accurately read weights after being tarred.

Temperature Sensor Precision

Value, Range, Tolerance, Units: Measures accurately to the nearest 1°C (1.8°F)

Test Results:

Test number	Medical thermometer reading (°F)	Temperature Sensor reading (°C)	Temperature Sensor reading converted (°F)	Percent difference (%)	Time to reach stable reading (s)
1	96.8	36	96.8	0	10
2	96.7	36	96.8	0.1	12
3	96.8	36	96.8	0	9

Table 3: Temperature sensor test part 1, compared against a medical thermometer. Both sensors were placed under a patient's mouth and the final temperatures recorded. Once removed, the recorded room temperature was recorded as shown in Table 4 below.

Test number	Initial room temperature reading (°C)	Final room temperature reading (°C)	Percent difference (%)	Time to reach stable reading (s)

1	21	21	0	8
2	21	21	0	13
3	21	21	0	11

Table 4: Temperature sensor test part 2, compared against a medical thermometer. Both sensors were placed under a patient's mouth and the final temperatures recorded as shown in Table 3 above. Once removed, the recorded room temperature was recorded as shown.

As seen in both the tables above, the precision of the temperature sensor is as accurate as we had hoped, reaching within 1°C of precision.
Changes: Initially our goal was to also test the precision at higher temperatures, but due to the coronavirus we were unable to attain a reasonable sensor to compare against in time.

Temperature Sensor Heat Resistance and Range
Value, Range, Tolerance, Units: Can withstand and measure temperatures ranging from 0°C up to 300°C (32°F - 575°F)
Test Results: Although not tested against another device, our sensor was able to withstand temperatures in an oven up to 450°F (the highest setting of the ovens used) and continue to display the temperature.
Changes: We were unable to procure an oven that reached 575°F before the coronavirus and so were unable to test the entire range as hoped. The lower end of the desired range was also moved from 250°F to 32°F to encapsulate a larger range of temperatures due to now being placed in food rather than just the oven.

RFID Tag Range
Value, Range, Tolerance, Units: All RFID tags should be readable when they are within 3 cm and directly in the line of sight of the back of the android device
Test Results: Every single NFC tag was able to be read within 3 cm of the back of the android device
Changes: We initially wanted a larger range for readability however settled on a range of 3 cm to ensure that the NFC tags were only scanned when intended.

RFID Measuring Cups Water Resistance
Value, Range, Tolerance, Units: All NFC tags attached to the bottom of the measuring cups and covered in a thin plastic film should maintain their structural integrity and readability after going through one cycle in the washing machine.
Test Results: Remains to be tested

Readability of NFC Tag with Mobile Application Download Link
Value, Range, Tolerance, Units: The NFC tag containing the mobile application download link should trigger the app download within 30 seconds of the tag being scanned. The brief introduction regarding the IoT Kitchen mobile application stored as a text to speech file should also be relayed to the user upon scanning the tag.
Test Results: Remains to be tested

Readability of NFC Tags on Measuring Cups
Value, Range, Tolerance, Units: A test will be conducted to check if the mobile application is able to interpret the NDEF data and register the specific measurement amount of the cup that is scanned. It will also be tested whether or not the mobile application is able to interpret the NDEF data from the NFC tag and make alterations to a recipe according to the specific cup that is scanned.
Test Results: Remains to be tested

Android Mobile Device
Value, Range, Tolerance, Units: Android supported device (Lollipop and above)
Test Results: All pages in the mobile application were successful. The user's profile name was shown upon logging in, which then was able to handle both bluetooth connection and the navigation bar. In addition, the front end of the mobile application was done to make the application user friendly and aesthetic. Lastly, all pages (login, recipe, sensor displacement, signout, user profile, and recipe list) are successful in doing their part.

Bluetooth Connection
Value, Range, Tolerance, Units: Stable bluetooth connection in a 5 meter range
Test Results: All segments were successful in both testings. The mobile application connected to the scale and showed the result on the bluetooth page. This was done so after the profile prompts users, asking if they wish to connect. After doing so, users will be connected to the scale and temperature sensor in which it will then send data from the objects to the mobile applcation's placement.

Data Storage
Value, Range, Tolerance, Units: After a cooking session, relevant information is successfully stored in the users database, including sensor data.
Test Results: Data is successfully stored in the database such as a reference to the recipe cooked and sensor data. References to the recipe cooked are always accurate as shown in tests, but sensor data is sometimes incorrectly stored due to the reasons mentioned below.
Changes: Not all of the data we had planned to save is currently being saved, such as date and times. We were in the middle of finishing this when instructed to stop working.

Navigation Bar

Value, Range, Tolerance, Units: Stable upon navigating throughout the application
Test Results: In prototype testing, all segments in the mobile application were able to be navigated without crashing. Users were easily able to navigate throughout and successfully go from one page to another. Users logged in, selected their recipe which navigated them to their scale and instruction page. Finally users were able to sign out.

Language Understanding

Value, Range, Tolerance, Units: English language understanding with 90% accuracy rate
Test Results: In prototype testing for prototypes 1 and 2, all test phrases were able to be matched to the correct intent with 100% accuracy. Through unofficial testing, we have found that the module will not crash even when provided with commands that it is not able to match to a specific intent. Because of these tests we can conclude that this requirement has been met without any changes required.

Major Changes

- Smart measuring cups became RFID tags due to feedback received.
- Decided to focus on one platform instead of multiple to begin due to client feedback.
- All of the below was being worked on at the time of being instructed to halt work due to the coronavirus
    - Addition and modification of recipes
    - Displaying of data
    - Search for and follow other users

7.0 Attachments

7.1 Appendix 1 – Technical Reference

[1] “Dialogflow Documentation,” Google, Mountain View, CA, USA, October 20, 2019 [Online]. https://cloud.google.com/dialogflow/docs/

[2] S. Al-Mutlaq and Alex, *Load Cell Amplifier HX711 Breakout Hookup Guide*, Sparkfun, July 22, 2016 [Online]. https://www.sparkfun.com/search/results?term=load+cell+amplifier

[3] Avia Semiconductor, “24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales” HX711 datasheet.

[4] Maxim Integrated, “Precision Thermocouple to Digital Converter with Linearization” MAX31856 datasheet.

[5] Johnson, Felicity. “Firebase: User Sign Up, Login & Data Management.” *Medium*, Medium, 9 Nov. 2016, [Online]. https://medium.com/@felicity.johnson.mail/firebase-user-sign-up-login-data-management-992d778b167

[6]“Android Bluetooth with Examples.” *Tutlane*, [Online]. www.tutlane.com/tutorial/android/android-bluetooth-with-examples.

7.2 Appendix 2 – Team Information

Name	Bio	Team Role
Addison Dolido addison@bu.edu	Addison Dolido is a senior studying Electrical Engineering. Addison has always been curious as to how the technology around him works and decided to learn how to create new devices. He has a broad range of technical experience including software development in C#, Python, Java, and Javascript, 3D modeling and printing, hardware design and integration, and even artificial intelligence and machine learning. Addison's passion for exercise and love to cook drew him to the IoT Kitchen project, while his desire to broaden his technical knowledge further caused him to take the role as hardware lead.	Hardware/ Software
Erin Dorsey edorsey1@bu.edu	Erin Dorsey is a senior studying Computer Engineering who hails from Southbury, CT. Erin has experience in software development in C/C++, C#, Python, and Javascript, in addition to her significant experience in project management honed by various school projects and internships. At one of these internships she was exposed to the development process of Natural Language Understanding agents, inspiring her to take the lead on this element of the IoT Kitchen project. Outside of class, Erin is pursuing a career in cybersecurity consulting. In her free time, she is an avid cook and baker.	NLU
Saransh Kothari saranshk@bu.edu	Saransh Kothari is currently a junior pursuing a degree in Electrical Engineering. His interests lie within the fields of artificial intelligence and quantum computing. Saransh left Hong Kong to begin his educational journey in Boston and started in the Questrom School of Business. After a year, he transferred into the College of Engineering to garner a technical skill set to supplement his strong background in business. Saransh is an Entrepreneur at heart and plans to delve deeper down this path upon graduation. Saransh is known as an ardent traveler who has visited over 50 countries thus far.	Hardware
Yuran Shi yuran@bu.edu	Yuran Shi is a senior Computer Engineering student with a minor in Chemistry. Yuran has experience in software development in C/C++, C# and verilog. Yuran also gained experience of using databases after internship at an insurance company. Yuran's experience with databases and passion for cooking inspired him to take part in the back-end development for the IoT kitchen mobile application. Outside school, Yuran is an advanced sommelier and decides to continually pursue higher certificates.	Software
Kenny Zheng ykzheng2@bu.edu	Kenny Zheng is a senior studying Computer Engineering with a concentration in Technology Innovation. Kenny gained interest in software development after dropping out from college as a freshman. His experience lies in C/C++, Python, HTML, CSS and JavaScript after working as a web developer, and teaching assistant. Inspired by software development, Kenny decided to take his prior experience and knowledge in his role as the front-end developer for the IoT Kitchen mobile application. Besides schoolwork, Kenny enjoys sleeping, eating, and more sleep. Kenny also enjoys long walks while jamming to old Tay Tay on Spotify. Kenny bought a typewriter for the COVID19 Pandemic.	Software